On a New Quantum Search Algorithm and Its Computational Complexity (New developments of generalized entropies by functional analysis)

AUTHOR(S):

Iriyama, Satoshi; Ohya, Masanori

数理解析研究所講究録
第1852巻 2013年 33-39

On a New Quantum Search Algorithm and Its Computational Complexity

S.Iriyama and M.Ohya
Tokyo University of Science

Abstract

In this paper, we introduce a new quantum algorithm for search problem, and discuss its computational complexity. This quantum algorithm is based on the OV quantum algorithm for SAT problem.

1 Introduction

Let X and Y be two finite sets and a function $f : X \to Y$. A search problem is to find $x \in X$ such that $f(x) = y$ for a given $y \in Y$. There are two different cases for the search problem: (S1) one is the case that we know there exists at least one solution x of $f(x) = y$ in X. (S2) The other is the case that we do not know the existence of such a solution. The second one is more difficult than the first one. S1 belongs to a class NP, however S2 does to a class NP-hard.

Since S2 contains S1 as a special case, we will discuss S2 only here. A search problem is defined by the following.

Problem 1 *(S2) For a given f and $y \in Y$, we ask whether there exists $x \in X$ such that $f(x) = y$.*

Without loss of generality for discrete cases, we take $X = \{0, 1, \cdots, 2^n - 1\}$ and $Y = \{0, 1\}$. Let $M_{f,X,Y}$ be a Turing machine calculating $f(x)$ and checking whether $f(x) = y$ with $x \in X$ and $y \in Y$. It outputs 1 when $f(x) = y$, 0 otherwise. To solve this problem, one can construct a Turing machine M_f running as follows:

Step1: Set a counter $i = 0$.

Step2: If $i > 2^n - 1$, then M_f outputs "reject", else calls $M_{f,X,Y}$ with the inputs $x = i$ and y, so that M_f obtains the result.

Step3: If the result of Step 2 is 1, then it outputs x.

Step4: If the result is 0, then it goes back to Step2 with the counter $i+1$.

In the worst case, M_f must call $M_{f,X,Y}$ for all x to check whether $f(x)=y$ or not, so that the computational complexity of the searching algorithm is the cardinal number of X.

In the sequel sections, we construct a quantum algorithm to solve the problem S2, and discuss on the computational complexity of it.

In the paper[8], we developed a new quantum algorithm for search problem, and showed that the computational complexity of it is polynomial of n. Moreover, we applied this quantum algorithm into prime factorization[9].

2 Quantum Searching Algorithm

From this section, we use a discrete function f. Let n be a positive number, and f a function from $X=\{0,1,\cdots,2^n-1\}$ to $Y=\{0,1\}$.

We show a quantum algorithm to solve the problem S2. To solve this problem, we denote x by the following binary expression

$$x=\sum_{k=1}^{n}2^{k-1}\varepsilon_k,$$

where $\varepsilon_1,\cdots,\varepsilon_n\in\{0,1\}$.

We divide the problem S2 into several problems as below. Here we start the following problem:

Problem 2 *Whether does there exist x such that $f(x)=1$ with $\varepsilon_1=0$?*

If the answer is "yes", namely $\varepsilon_1=0$, then there exists at least one $x=0\varepsilon_2\cdots\varepsilon_n$ such that $f(x)=1$. If the $\varepsilon_1\neq 0$, then one considers two cases; the $\varepsilon_1=1$, or there does not exist any x such that $f(x)=1$.

We go to the next problem with the result of the above problem:

Problem 3 *Whether does there exist x such that $f(x)=1$ with $\varepsilon_2=0$ for the obtained ε_1?*

After solving this problem, we know the value of ε_2, for example, when $\varepsilon_2=0$, x is written by $00\varepsilon_3\cdots\varepsilon_n$ or $10\varepsilon_3\cdots\varepsilon_n$.

Furthermore, we check the ε_i, $i=3,\cdots,n$ by the same way as above using the information of the bits from ε_1 to ε_{i-1}. We run the algorithm from ε_1 to

ε_n, and we look for one x satisfying $f(x) = 1$. Finally in the case that the result of the algorithm is $x = 1 \cdots 1$, we calculate $f(1 \cdots 1)$ and check whether $f(1 \cdots 1) = 1$ or not. We conclude that (1) if it becomes 1, $x = 1 \cdots 1$ is a solution of search problem, and (2) otherwise, there does not exist x such that $f(x) = 1$.

Let m be a positive integer which can be written by a polynomial in n. Let $\mathcal{H} = \left(\mathbb{C}^2\right)^{\otimes n+m+1}$ be a Hilbert space. The m qubits are used for the computation of f, and the dust qubits are produced by this computation. When f is given, we can fix m. We will show in the next section that this algorithm can be done in a polynomial time.

We construct the following quantum algorithm $M_Q^{(1)}$ to solve the problem 2. Let $\left|\psi_{in}^{(1)}\right\rangle = |0^n\rangle \otimes |0^m\rangle \otimes |0\rangle \in \mathcal{H}$ be an initial vector for $M_Q^{(1)}$, where the upper index (1) comes from the quantum algorithm checking the bit ε_1. The last qubit of $\left|\psi_{in}^{(1)}\right\rangle$ is for the answer of it, namely "yes" or "no". If the answer is "yes", then the last qubit becomes $|1\rangle$, otherwise $|0\rangle$.

The quantum algorithm $M_Q^{(1)}$ is given by the following steps. We start $M_Q^{(1)}$ with $\varepsilon_1 = 0$.

Step1: Apply Hadamard gates from the 2nd qubit to the n-th qubit.

$$I \otimes U_H^{\otimes n-1} \otimes I^{m+1} \left|\psi_{in}^{(1)}\right\rangle = \frac{1}{\sqrt{2^{n-1}}} |\varepsilon_1 (= 0)\rangle \otimes \left(\sum_{i=0}^{2^{n-1}-1} |e_i\rangle \right) \otimes |0^m\rangle \otimes |0\rangle$$
$$= \left|\psi_1^{(1)}\right\rangle$$

where $|e_i\rangle$ are

$$|e_0\rangle = |0 \cdots 0\rangle$$
$$|e_1\rangle = |1 \cdots 0\rangle$$
$$\vdots$$
$$|e_{2^{n-1}-1}\rangle = |1 \cdots 1\rangle$$

Let U_f be the unitary operator on $\mathcal{H} = \left(\mathbb{C}^2\right)^{\otimes n+m+1}$ to compute f, defined by

$$U_f |x\rangle \otimes |0^m\rangle \otimes |0\rangle = |x\rangle \otimes |z_x\rangle \otimes |f(x)\rangle$$

where z_x is the dust qubit produced by the computation.

Step2: Apply the unitary operator U_f to the state made in Step1, and store the result in the last qubit.

$$U_f \left|\psi_1^{(1)}\right\rangle = \frac{1}{\sqrt{2^{n-1}}} |0\rangle \otimes \left(\sum_{i=0}^{2^{n-1}-1} |e_i\rangle \otimes |z_i\rangle \otimes |f(0e_i)\rangle \right)$$
$$= \left|\psi_2^{(1)}\right\rangle$$

where z_i is the dust qubits depending on e_i.

Step3: We take the last qubit by the projection from the final state $\left|\psi_2^{(1)}\right\rangle$ such that

$$(1-p)|0\rangle\langle 0| + p|1\rangle\langle 1| = proj.\left|\psi_2^{(1)}\right\rangle\left\langle\psi_2^{(1)}\right|$$

where $p = card\{x|f(x)=1, x=0\varepsilon_2\cdots\varepsilon_n\}/2^{n-1}$.

Step4: After the above formula, the state is a pure state or a mixed state. If the state is mixed and $p \neq 0$ however very small, then apply the Chaos Amplifier given in the Appendix to check whether the last qubit is in the state $|1\rangle\langle 1|$. If we find that the last qubit is in the state $|1\rangle\langle 1|$, then $p \neq 0$, which implies that there exists at least one solution of $f(x)=1$ for $\varepsilon_1 = 0$. If we do not find that the last qubit is in the state $|1\rangle\langle 1|$, namely $p=0$, then there are two possibilities that are $\varepsilon_1 = 1$ or no solutions $x \in X$ of $f(x)=1$.

After this algorithm, we know that if $\varepsilon_1 = 0$ or 1, then the last qubit is 1 or 0, respectively. We write this process as $M_Q^{(1)}(0^n) = \varepsilon_1$ where 0^n means the initial vector.

Next we modify Step1 of the algorithm $M_Q^{(1)}$ as:

Step1: Apply Hadamard gates from 3rd qubit to n-th qubit.

And we call this algorithm $M_Q^{(2)}$. The index (2) means that the algorithm check ε_2. We start $M_Q^{(2)}$ with the initial vector $\left|\psi_{in}^{(2)}\right\rangle = |\varepsilon_1, 0^{n-1}\rangle \otimes |0^m\rangle \otimes |0\rangle$ instead of $\left|\psi_{in}^{(1)}\right\rangle$.

So forth we obtain the bit ε_2, and write as $M_Q^{(2)}(\varepsilon_1, 0^{n-1}) = M_Q^{(2)}\left(M_Q^{(1)}(0^n), 0^{n-1}\right) = \varepsilon_2$.

In generally, we write the algorithm $M_Q^{(i)}(\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{i-1}, 0^{n-i+1})$ for an initial vector $\left|\psi_{in}^{(i)}\right\rangle = |\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{i-1}, 0^{n-i+1}\rangle \otimes |0^m\rangle \otimes |0\rangle$ as the following:

Step1: Apply Hadamard gates from $i+1$-th to n-th qubits.

$$I^{\otimes i} \otimes U_H^{\otimes n-i} \otimes I^{m+1} \left|\psi_{in}^{(i)}\right\rangle = \frac{1}{\sqrt{2^{n-i}}} |\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{i-1}\rangle \otimes \left(\sum_{k=0}^{2^{n-i}-1} |e_k\rangle \right) \otimes |0^m\rangle \otimes |0\rangle$$
$$= \left|\psi_1^{(i)}\right\rangle$$

Step2: Apply the unitary gate to compute f for the superposition made in Step1, and store the result in $n+m+1$-th qubit.

$$U_f \left|\psi_1^{(i)}\right\rangle = \frac{1}{\sqrt{2^{n-i}}} |\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{i-1}\rangle \otimes \left(\sum_{k=0}^{2^{n-1}-1} |e_k\rangle \otimes |z_k\rangle \otimes |f(\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{i-1}, e_k)\rangle \right)$$
$$= \left|\psi_2^{(i)}\right\rangle$$

Step3: Take the last qubit by the projection from the final state $\left|\psi_2^{(i)}\right\rangle$ such that

$$(1-p)|0\rangle\langle 0| + p|1\rangle\langle 1| = proj.\left|\psi_2^{(i)}\right\rangle\left\langle\psi_2^{(i)}\right|$$

Step4: Apply the Chaos Amplifier to find that the last qubit is $|1\rangle\langle 1|$.

After this algorithm $M_Q^{(i)}\left(\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{i-1}, 0^{n-i+1}\right)$, we know the bit ε_i such that $f(x)=1$. Each $M_Q^{(i)}$, $i \geq 2$ use the result of all $M_Q^{(j)}$ $(j<i)$ as an initial vector.

3 Computational Complexity of the Quantum Searching Algorithm

Here, we calculate the computational complexity of the quantum searching algorithm. The computational complexity is the number of the total unitary gates and amplification channels in our search algorithm.

In the above section, the quantum algorithm for binary search is given by the products of unitary gates denoted by U_i below. Let $\left|\psi_{in}^{(i)}\right\rangle$ be an initial vector for the algorithm $M_Q^{(i)}$ as

$$\left|\psi_{in}^{(i)}\right\rangle = \left|\varepsilon_1 \cdots \varepsilon_{i-1}, 0^{n-i}\right\rangle \otimes |0^m\rangle \otimes |0\rangle,$$

and it goes to the final vector

$$U_i \left|\psi_{in}^{(i)}\right\rangle = \frac{1}{\sqrt{2^{n-i}}} \left|\varepsilon_1, \cdots, \varepsilon_{i-1}\right\rangle \otimes \left(\sum_{k=0}^{2^{n-1}-1} \left|e_k\right\rangle \otimes \left|z_k\right\rangle \otimes \left|f\left(\varepsilon_1, \cdots, \varepsilon_{i-1}, e_k\right)\right\rangle \right)$$
$$= \left|\psi_2^{(i)}\right\rangle$$

where $f(\varepsilon_1, \cdots, \varepsilon_{i-1}, e_k)$ is the result of the objective function for searching. The above unitary gates U_i for the algorithm M_i is defined by

$$U_i = U_f \left(U_H\right)^{\otimes n-i} \prod_{\{x_k | x_k=1\}} U_{NOT}(k)$$

where $U_{NOT}(k)$ is to apply a NOT gate for the k-th qubit only when the result of stage k is 1, $(k = 1, 2, \cdots i-1)$.

The computational complexity T of the quantum binary search algorithm $T(U_n)$ is given by the total number of unitary gates and quantum channels for the amplification. We obtain the following theorem[8].

Theorem 4 *We have*

$$T = \frac{13}{8}n^2 - \frac{9}{4}n + nT(U_f)$$

where $T(U_f)$ is a given complexity associated to the function f.

References

[1] L.A.Levin, Universal sequential search problems. Problems of Information Transmission, 9(3):265–266, 1973

[2] L.A.Levin, Randomness Conservation Inequalities: Information and Independence in Mathematical Theories. Information and Control, 61:15-37, 1984.

[3] R. J. Solomonoff, Optimum sequential search. Memorandum, Oxbridge Research, Cambridge, Mass., June 1984.

[4] P.W.Shor, Algorithms for quantum computation: discrete logarithms and factoring, Proc. of 35th Annu. Symp. on Fou. of Comp. Sci., IEEE Press, Los Alamitos, CA, 1994

[5] L.K.Grover, A fast quantum mechanical algorithm for database search, Proceedings, 28th Annual ACM Symposium on the Theory of Computing, p. 212, 1996

[6] M.Ohya and I.V.Volovich (2003) New quantum algorithm for studying NP-complete problems, Rep.Math.Phys.,52, No.1,25-33

[7] M.Ohya and I.V.Volovich (2003) Quantum computing and chaotic amplifier, J.Opt.B, 5,No.6 639-642

[8] S.Iriyama, M.Ohya, I.V.Volovich, On Quantum Algorithm for Binary Search and Its Computational Complexity, TUS preprint, 2012

[9] K.Goto, S.Iriyama, M.Ohya, I.V.Volovich, On Quantum Algorithm of Prime Factoring Using Quantum Binary Search, TUS preprint, 2012